To appear in: Vallverdú, J. & Casacuberta, D. (Eds.) (2010). *Handbook of Research on Synthetic Emotions and Sociable Robotics: New Applications in Affective Computing and Artificial Intelligence.* Hershey, Pennsylvania: IGI Global.

Emotional Memory and Adaptive Personalities

Anthony G. Francis, Jr.

Google
1600 Amphitheatre Parkway
Mountain View, CA 94043
centaur@google.com

Manish Mehta

College of Computing,
Georgia Institute of Technology
Atlanta, GA 30332
mehtama1@cc.gatech.edu

Ashwin Ram

College of Computing,
Georgia Institute of Technology
Atlanta, GA 30332
ashwin@cc.gatech.edu

Abstract

Believable agents designed for long-term interaction with human users need to adapt to them in a way which appears emotionally plausible while maintaining a consistent personality. For short-term interactions in restricted environments, scripting and state machine techniques can create agents with emotion and personality, but these methods are labor intensive, hard to extend, and brittle in new environments. Fortunately, research in memory, emotion and personality in humans and animals points to a solution to this problem. Emotions focus an animal's attention on things it needs to care about, and strong emotions trigger enhanced formation of memory, enabling the animal to adapt its emotional response to the objects and situations in its environment. In humans this process becomes reflective: emotional stress or frustration can trigger re-evaluating past behavior with respect to personal standards, which in turn can lead to setting new strategies or goals. To aid the authoring of adaptive agents, we present an artificial intelligence model inspired by these psychological results in which an emotion model triggers case-based emotional preference learning and behavioral adaptation guided by personality models. Our tests of this model on robot pets and embodied characters show that emotional adaptation can extend the range and increase the behavioral sophistication of an agent without the need for authoring additional hand-crafted behaviors.

Introduction

When we see a pet we've met before, we recall not just its name and temperament but how our interactions with it made us feel. We feel happy when we see the dog we had fun playing with, and feel sour about the cat that shocked us with its hiss. And just as we learn from them, they learn from us; the dog, remembering its happiness upon playing with us, may seek us out when we are down; and the cat, remembering our shocked reaction when it hissed, may avoid us, or be more cautious with its anger in the future. Pets don't need to be 'configured' to live with us, and neither do we: all we need is the ability to react emotionally to our situations, a memory for our past emotional states, and the ability to let those recalled emotions color our current emotional state and guide our behaviors appropriately. We argue that robots and synthetic characters should have the same ability to interpret their interactions with us, to remember these interactions, and to recall them appropriately as a guide for future behaviors, and we present a working model of how this can be achieved.

Of course, humans are more complicated than pets; we have not just emotions but also ideals for our behavior, and can modify our reactions and plans when they violate our ideals. We may snarl back at the hissing cat, but that outburst of emotion can make us reconsider when we should show anger. Even if we do not reconsider at first, if we see the same cat multiple times we may eventually be prompted to figure out why it continues to try to enter our new home, to realize it was probably abandoned, and to change our routines to leave food for it – turning a hissing cat into a new companion. It may seem a tall order make robots have this kind of flexibility – but we argue it is possible by using emotion to trigger behavior revision guided by a personality model, and we present a working model of how it can be achieved.

In this chapter, we review efforts to build agents with believable personalities, point out problems particular to making these personalities convincing over long-term interactions with human users, and discuss research in cognitive science into the mechanisms of memory, emotion, and personality. Based on these psychological results, we present a method for building believable agents that uses emotion and memory to adapt an agent's personality over time. We then present two case studies illustrating this idea, the first demonstrating emotional long term memory in a robot, and the second demonstrating emotion-driven behavioral updates in an embodied character. Finally, we conclude with lessons learned.

Background

What Kind of Agents Need Memory, Emotion, and Personality Models?

In our work we are interested in affective systems: robots and agents designed to display, respond to, or make use of emotional states, in particular those which interact with humans over a long period of time in relatively unconstrained settings. Research into what makes characters appear believable indicates that changing and expressing emotion are key to maintaining believability over time (Loyall, 1997). We argue that using explicit emotion models integrated into an agent's memory but guided by the agent's personality model can aid the development of agents for long term interaction; to explain why, we will briefly review some popular techniques for creating believable agents and some typical problems that can arise.

Techniques for Creating Agent Personalities

Entertainment robots and embodied characters typically are designed to have distinctive personalities which affect how they behave towards their human user or how they act within the virtual world of a game. At first blush, such agents do not need a complex model of emotion: instead, simple reactions can trigger expressive behaviors that communicate to a user the impression that they are interacting with a real character with an inner emotional life (Johnston and Thomas 1995, Paiva 2005, Standifer 1995). For example, a simple finite state automaton could be used to make an enemy approach, make threatening gestures until wounded, and then scream in fear and run away.

While finite state automata are easy to develop, they are predictable. Over long game sessions, a character's static behavioral repertoire may result in repetitive behavior that hurts believability (Saltzman 1999). Therefore, many developers of computer games and robotic toys have turned to hierarchical, probabilistic and fuzzy state machines (Schwab 2004). The advantage of these systems is that layered control supports sophisticated behaviors, and probabilistic transitions makes the actual behavior of the agent nondeterministic, less predictable, and more realistic.

Other game developers have turned to scripting languages which allow arbitrarily sophisticated behaviors (Millington 2006). However, creating characters using scripting languages or fuzzy state machines can be very labor intensive. Computer game manufacturers typically employ dozens of artists and engineers to perfect very simple characters (for examples, see the Postmortems in Game Developer Magazine, e.g. Huebner 2000, Spector 2000, Ohlen et al 2001, etc.).

Challenges in Creating Agent Personalities

When authoring a character's behavior set, it is hard to imagine and plan for all possible scenarios it might encounter; despite extensive play-testing, any character may ultimately "break". Incorporating knowledge representation and planning techniques can enable an agent's behavior to become more flexible in novel situations, but this too can require extensive programming effort (Mateas and Stern, 2003), and it does not guarantee success: when an agent's behavior fails to achieve their desired purpose, most characters are unable to identify the failure and will continue the ineffective behavior.

A few game developers have tried incorporating explicit emotional models to improve the believability of their agents, which can effectively communicate to a user the impression that they are interacting with a real character with an inner emotional life (Reilly 1996). Notably, the characters in *The Sims* (Maxis 2000) incorporated explicit "motivational" states, which could be satisfied by objects in the environments which were labeled with the drives that they could satisfy. This system meant that a character who was hungry would naturally turn to a nearby refrigerator without the need for explicit programming of that behavior (Woodcock 2000b).

Another solution is to make characters adapt, but game developers tend to avoid adaptive agents because it makes playtesting difficult and debugging user problems nearly impossible (Woodcock 2000a, Tozour 2002). There are exceptions; notably, the characters in *Creatures* (Cyberlife 1997) could be trained through emotional interactions; however, these characters had deliberately limited lifespans to encourage player identification with them, so we cannot say how this model would have fared over longer periods (Champanard 2007).

Exploiting Adaptation for Better Agents

Based on these tradeoffs – simpler systems are easy to develop but predictable and brittle, more complicated agents are more flexible but harder to author, and adding adaptation to agents makes them more convincing but less controllable – we argue that a better way to achieve long-term believability is to introduce explicit models of emotion, memory and personality control, based on how real human and animals emote, adapt, and maintain their personalities.

The approach we take is cognitively grounded artificial intelligence: we start from the premise that constructs like memory, emotion and personality are real phenomena that exist in humans, and that to emulate them we need to understand precisely what these phenomena are. However, once that understanding is established, development of artificial intelligence control systems for agents may simplify these psychological models significantly in an attempt to make the implementation of these systems easier and more efficient.

Theories of Memory, Emotion and Personality in Cognitive Science

The Nature of Emotion

Current psychological theories model emotion in humans and animals as three interrelated processes: physiological

responses, overt behaviors, and conscious feelings that occur in response to events of potential relevance (Gluck 2008); many researchers postulate that the functional role of emotions is to determine what problems are currently important and to focus the agent's mind and actions on them (e.g., LeDoux 1996, Damasio 2000, Minsky 2007). Simon (1983) points out that the environment presents many challenges — food, safety, sex, etc. — that cannot all be met at once, and that emotions provide a way to shift our focus, allowing us to drop our search for food in the face of imminent death within the jaws of a tiger.

Frijda (1987) argues along similar lines that agents have a variety of concerns, and when threats to our concerns become pressing enough, they interrupt our thoughts and actions — even if we have not been attending to them (e.g., Ohman et al 2000, Winkielman and Berridge 2004, Ruys & Stapel 2008). This can cause an immediate behavioral response, but more importantly it can lead the agent to change its stance towards the environment, switching to a new mode of behavior that leads to the selection of new goals, plans, and actions.

Memory and Learning

Memory refers to an agent's capacities to modify its behavior based on experience. There is not enough space for us to discuss the extensive literature of memory retrieval; for more information see (Anderson 2000, Tulving & Craik 2000, Purdy et al. 2001, Gluck 2008). Types of memory include *procedural memory* for skills and *declarative memory* for facts, which includes *associative memory* connecting stimuli with responses, *semantic memory* for general concepts divorced from specific stimuli, and *episodic memory* for when and where specific facts were learned (for a review see McGaugh 2003).

Episodic memories are not taken as static "snapshots": most of us are familiar of with learning a friend's new phone number, only to forget it later and dial their old number. Under normal circumstances, memories are consolidated over a period that takes somewhere between hours and decades (McGaugh 2007, Haist et al. 2001). Furthermore, memories are reconstructive, continually evolving as missing details are filled in on recall (Bartlett 1932).

The Relationship of Memory and Emotion

Memories are affected by emotion: we often have heightened recollection of events associated with charged emotional events like national disasters. Even very mildly emotional events, such as reading emotionally charged stories, can create memories stronger and more detailed than memories of emotionally neutral events (McGaugh 2003, Gluck et al 2008), with the caveat that when emotional stress becomes so great that it interferes with cognitive functioning, memory performance drops off again (Benjamin et al. 1981).

Emotion's effect on memory appears to begin when brain structures responsible for emotion, such as the amygdala, detect an emotional event and communicate with brain structures responsible for consolidating short term memory into long term memory, such as the hippocampus. When the emotion centers detect emotional events, they simultaneously excite nearby brain regions and release stress hormones that prepare the body for action. These stress hormones feed back into the emotion centers of the brain themselves, and the combination of the initial excitation and the subsequent feedback strengthens the formation of memories in the consolidation regions (McGaugh 2003, Gluck et al. 2008).

Personality and Self-Regulation

Personality refers to distinctive yet stable individual differences in behavior and cognition. For our purposes we are most interested not just in the content of a person's personality, but in the self-regulation mechanisms by which they maintain it. When we choose not to eat that cookie because we're on a diet, congratulate ourselves on successfully abstaining, or berate ourselves for eating the second one, we are simultaneously comparing our performance with respect to a standard (a cognitive act) and evaluating how we measure up in a positive or negative way (an emotional act). Based on how good we are at our diets (our prospects of success), we may decide to eat no more than one cookie a week (setting a goal).

These four elements – personal standards which can be used to evaluate our behavior, emotional evaluation of our actions with respect to our standard, evaluating our prospects of success, and setting goals — appear key to our regulation of our behavior (Caprara & Cervone 2000, Minsky 2007). However, we are not constantly evaluating our own performance; sometimes it takes a disappointing surprise on the scale to realize we've failed at our diet. Self-regulation can lead to changing our goals, re-evaluating our prospects of success, or even altering our standards; but it is also a *conscious* process which is more likely to engage when we experience some disruption, such as failing at a task, receiving feedback, or becoming socially self-conscious (Caprara & Cervone 2000).

Implementation of Memory, Emotion and Personality Models in Artificial Intelligence Systems

These results seem to show that emotion plays a key role not just in alerting us about the need for action, but also in determining what we remember and even when we should change. But these psychological results do not directly translate into control systems for an affective robot or an embodied character; to take advantage of these results we must exploit (or develop) implementable artificial intelligence models of memory, emotion and personality.

Modeling Emotion in Intelligent Systems

Two popular emotion models in artificial intelligence are multiple concerns models and cognitive evaluation models.

Multiple concerns models map emotion to competing systems detecting events of importance to an agent. Features in perception or cognition act as triggers for concerns, which in turn can trigger emotions, or behavioral modes, that change how the agent selects its behaviors. The PEPE system we describe later was a multiple concerns model based on ideas from

Simon (1983) and Frijda (1987). A similar model was implemented by Velasquez (1997, 1998) based on a synthesis of ideas from Ekman (1992), Izard (1991), and Johnson-Laird and Oately (1992).

Cognitive evaluation models map emotions to a hierarchy of evaluations of physical and mental conditions that can affect an agent. Ortony, Clore and Collins proposed a cognitive evaluation model (1988) in which emotions consist of valenced reactions to the outcome of events, the actions of agents, and the properties of objects, especially with respect to how those outcomes, actions and properties affect us and whether or not we know whether these potential effects have been realized. The OCC model was deliberately designed to be implementable on an artificial intelligence system: the emotion model of the ABL system we describe later was derived from the Em (Reilly 1996) implementation of OCC, and many other systems have used related models (e.g., Elliott 1992, Studdard 1995, Koda 1996, Karunaratne & Yan 2001, Bartneck 2002, Li et al. 2007).

Memory Retrieval and Machine Learning

There are many machine learning techniques that can exploit past experience (Mitchell 1997, Alpaydin 2004); two popular techniques are case-based reasoning and reinforcement learning.

A *case based reasoning* system learns by storing specific episodes, or cases; this involves deciding what parts of a current experience to store, what lesson the experience teaches, and how to label the experience for retrieval later (Kolodner 1993). Once an agent has retrieved a case from its case library, it must adapt it to solve the current problem, and then ideally update its records of its experiences based on the new outcome (Kolodner 1984).

Reinforcement learning, commonly used in robotic learning and control, attempts to find a scheme for selecting actions, or *policy*, that maximizes the agent's return on a reward function (Sutton & Barto 1998). Reinforcement learning and case-based reasoning can be combined: the LARC model (Santamaria 1997) stores cases that trace the recent history of the agent's perceptions and behaviors along with the accompanying reward function; these cases are retrieved later using an adaptive nearest-neighbor approach.

Personality Regulation and Behavior Transformation

The behavior of an agent can be considered a reactive plan, making revising an agent's behavior a problem of runtime reactive-plan revision. There are many approaches to revise plans on failure.

Classical planning assumes the agent is the sole source of change, actions are deterministic and sequential, and that the world is fully observable. *Conditional planners* such as Conditional Non Linear Planner (Peot and Smith, 1992) and Sensory Graph Plan (Weld et. al, 1998) support sensing actions so that the appropriate conditional branch of the plan is taken, but this can cause plans to grow exponentially.

Decision-theoretic planners deal well with exogenous events and non-determinism by modeling problems as Markov decision processes (MDPs) and solving problems by learning a policy, like reinforcement learners. *Partially observable MDPs* can even be used when the world is not fully observable. But these approaches require many iterations to converge, their state spaces become intractable in complex domains, and they learn static policies that cannot be easily retrained during dynamic game play.

Transformational planning handles these problems of complexity and non-determinism by not reasoning about the problem domain to generate a new plan, but instead by reasoning about the failing plan *itself*, transforming it to fix the failure without breaking the rest of the plan.

Emotional Long Term Memory for Configuration and Personality

As an alternative to putting vast amounts of authoring effort into designing behaviors that handle every situation, we propose to create believable, engaging artificial characters capable of long-term interaction with a human user by explicitly modeling the emotional adaptation that goes on in humans and animals. Our model includes the simulation of emotional response to engage a human user's interest, adaptation of that response in a naturalistic way to maintain the user's interest, and ongoing monitoring of how that adaptation fits the agent's personality to avoid violating the human's expectations.

- **Reactive Control** makes it possible to author sophisticated behaviors on a robot or embodied character constantly interacting with a changing real-time environment. While not strictly part of our emotion and personality model, both implemented case studies used a reactive control system that supported complex behaviors as the platform to build the emotion, adaptation and personality model systems that comprise the rest of our model.
- **Emotional responses** allow an agent to respond appropriately to its environment before learning. For example, even if a person has never seen a tiger, if a tiger appears and roars, that person will become afraid and run away. We model emotional response as situation evaluation, stance selection, and behavior selection: the agent evaluates the appearance of a large, loud animal as a threat to safety, chooses a stance towards this new object that it should be avoided, and on the basis of this stance selects good avoidance behaviors like running.
- **Emotional adaptation** allows an agent to analyze the results of an experience and alter its future behavior. The next time that the unfortunate person of our previous example sees a tiger, he will not need to wait for the tiger to roar to decide to run. We model this in terms of object recognition, outcome association, and response adaptation: the agent identifies the tiger as a distinct object, stores it in memory associated with the threat to safety and the

stance of fear, and adapts its situation evaluation process when the object is seen again, enabling him to select fight or flight behaviors faster than before.

- **Personality modeling** enables an agent to remain consistent over time. As situations change and emotions adapt, certain aspects of the behavior of a human or an animal tend to stay the same; we describe these as the agent's personality. We model this in terms of emotional monitoring, personality evaluation, and behavioral transformation: the agent monitors that it has been frequently frightened on this path, realizes that it does not like being repeatedly frightened, and changes its behavior to avoid the path.

This model allows an agent to learn from its basic emotional responses, to expand its repertoire of emotional behavior, and to guide its emotional development with respect to its personality. More importantly, from our perspective, it provides a natural interface between man and machine, one that allows an artificial pet or virtual character to learn its user's preferences automatically, without explicit configuration or programming on the part of the user, while still remaining consistent with the original author's design intent. We now describe case studies implementing two aspects of this model.

Case Study: Emotional Long Term Memory for Agent Configuration in the PEPE Project

Most users would probably not want to crack open a manual just to tell a robot pet not to disturb their afternoon nap; they would rather communicate this simply through natural language, or even more simply, by expressing pleasure or displeasure at the agent's behavior. This is the *configuration problem*: how can humans express their preferences towards a synthetic character? Our solution to this problem is *emotional long term memory*, an approach which proposes that an agent can be configured to interact appropriately with humans by giving the agent an emotional model, a memory for those emotions, and the ability for remembered emotions to affect its current state.

In our motivating example, pets learn their owner's preferences by remembering and recalling the emotional experiences that resulted from their past behaviors. It would not be necessary to "configure" such a robot not to hop up onto a table: it would only be necessary to loudly shout "down" and let the robot learn to associate hopping up on a table with painful emotional shocks. Our model can also extend this idea to multiple agents: a pet can learn to avoid the crotchety grandparent and to approach the playful child simply by having "normal" emotional reactions to the grandparent being crotchety and the child being playful.

We tested these ideas for emotion-based configuration in the Personal Pet (PEPE) Project, a joint project between Georgia Tech and Yamaha Motor Corporation to produce a believable, engaging artificial pet capable of long-term interaction with multiple human users. The PEPE project developed a cognitively inspired robot control architecture that made it easy for us to author complex behaviors on top of a reactive system; on top of this we implemented a model of emotional long term memory in which memories of emotional experience influenced future emotional responses.

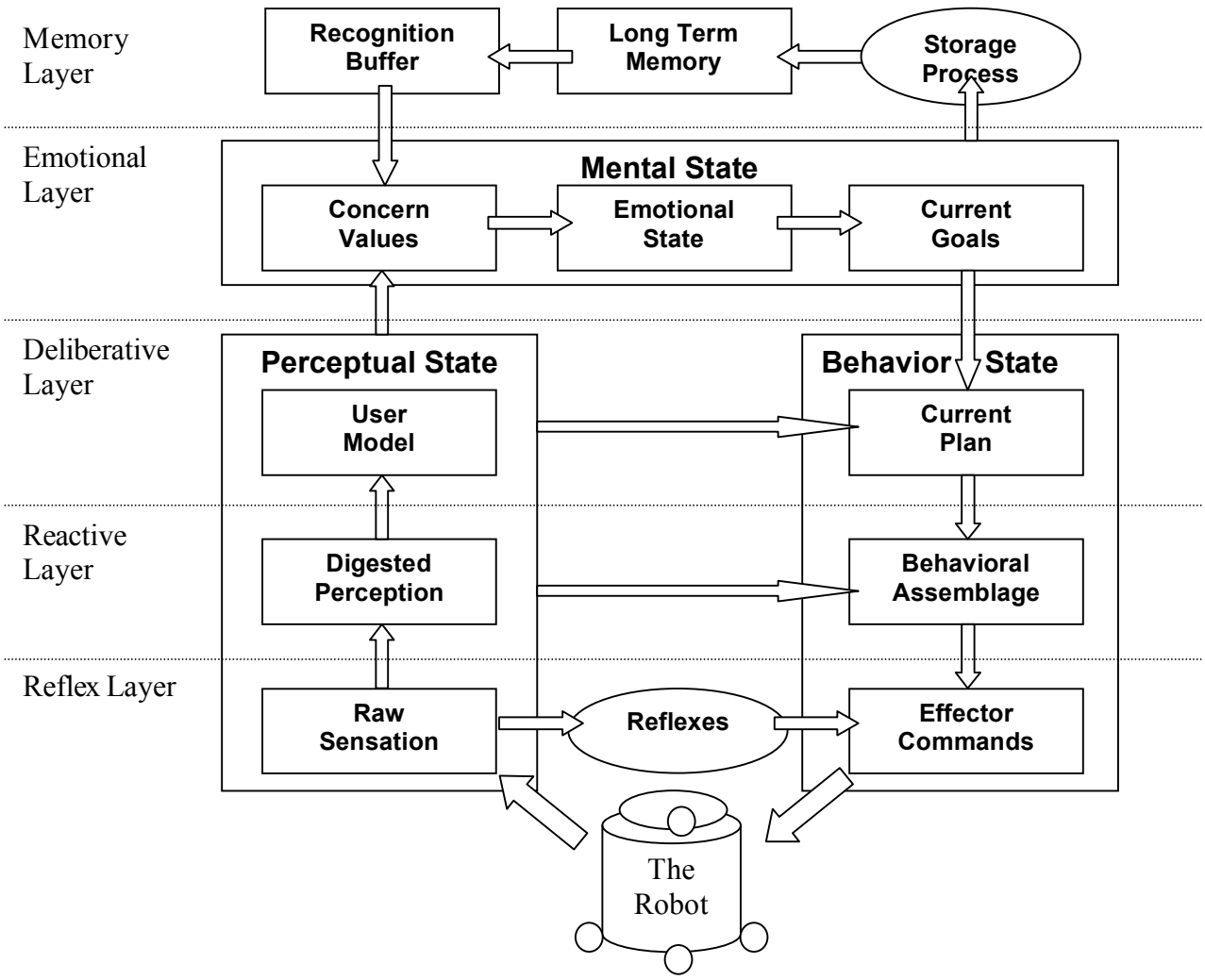

Figure 1. Layers of control in the PEPE architecture

Reactive Control using the PEPE Architecture

The PEPE architecture supports several overlapping goals: controlling a robot robustly, performing sophisticated behaviors, and adapting in response to user behaviors. To achieve this, the PEPE architecture uses a layered approach in which complex cognitive processing is layered atop simpler, faster behaviors. These layers include reflexive, reactive, deliberative, emotional and memory processes (Figure 1):

- **The Reflex Layer** directly connects raw sensation to effector commands. Reflexes are high-priority behaviors which must be executed immediately to be effective, such as emergency obstacle avoidance and shutting down electric motors if they begin to draw too much current.
- **The Reactive Layer** maps processed perceptions to effector commands in a fast constant-time fashion using *motor schemas* (Arkin 1989), which produce an output vector encapsulating a simple behavior, such as obstacle avoidance, noise or pursuit of goals. Motor schemas can be summed through fast vector addition to produce coherent and consistent reactive behavior. Sets of motor schemas can be combined into a *behavior configuration* which determines robot heading, camera pan and tilt, and other effector commands.
- **The Deliberative Layer** combines a planner with a plan execution system that activates behavior configurations in the reactive level based on the current state of an executing plan. Each state of a plan defines a behavior configuration designed to achieve or maintain some state, akin to an operator in a traditional plan. When preconditions for a state change are met, the planner swaps in the appropriate behavior configuration — ensuring *some* behavior configuration is active regardless of how much effort is expended constructing plans.

- **The Emotional Layer** computes impacts to the agent's concerns, resulting in an emotional state that influences lower levels primarily through the setting of goals. This contributes to robustness: if no emotion arises or an emotion adds a goal for which there is no current plan, the agent can continue executing its current plan and behavior configuration until the planner decides what it needs to do.
- **The Memory System** records rewards and preferences on events and objects based on the agent's complete mental state, including perceptions, plans, goals and emotions. Its influence on lower levels in the current architecture is primarily by feeding back into the emotional layer, again contributing to robustness; if the agent is in a novel situation it falls back on its basic emotional responses.

Each layer in this architecture is influenced by information from layers above it without depending on it. In this way, more abstract and symbolic layers can influence lower layers without disrupting the flow of their processing, no matter how much time is required to perform that symbolic computation. This combines aspects of subsumption architectures (Brooks 1986) and high-level symbolic systems to produce sophisticated and robust behavior even in the face of noisy sensor data.

User Adaptation using Emotional Long Term Memory

ELTM was intended to both enable the adaptation of the emotional response in a naturalistic way that would maintain a single user's interest over the life of the pet, and to enable the robot to 'configure' itself with respect to different members of a household that showed it different levels of affection. Basic (non-learned) emotional response in ELTM is composed of three major functions: situation evaluation, stance selection, and goal selection, which correspond directly to the concerns, emotions, and goals of Frijda's theory:

- **Situation Evaluation:** The agent constantly evaluates the experiences it is having with respect to the concerns that it has: this produces a vector of evaluations about the agent's safety, socialization, food, battery level, and so on. This vector represents roughly the agent's basic emotional appraisal of a situation.
- **Stance Selection:** As situations change, the agent may choose a s*tance* towards its current situation: this "emotional stance" consists of a high-level behavioral mode relevant to the active concerns, such as approaching, fleeing, investigating and so on. This roughly represents the agent's emotional response to its situation.
- **Goal Selection:** When an agent changes its behavioral mode, it must select goals and actions consistent with this new emotional stance, which may mean modifying the current plan or replacing it entirely. This roughly corresponds to what the agent's emotional response is motivating it to do.

Adapting an emotional response in this model consists of modifying how the agent evaluates its situations, selects stances, and selects goals. ELTM makes these modifications using an object identification system and a long term memory retrieval system integrated with the emotion model. Emotional adaptation was achieved with three processes: recognizing situations, associating them with outcomes, and responding to these emotional associations:

- **Recognizing Situations:** The agent has an object identification system which attempts to identify distinct entities in the agent's environment, paired with a long term memory system which stores records of what objects have been previously seen as well as specific situations in which they have occurred. Every experience in long term memory is labeled with a vector of features. When a new experience arises, the agent searches for the most similar past experience — the past experience that shares the most features with the current experience.
- **Associating Outcomes:** When an emotionally significant event happens – a large change in the situation evaluation, or a shift in the selected stance - the agent stores the current experience tagged with the emotional vector as outcome information. This is essentially a case in a case based reasoning sense, storing the "lesson" that this experience has taught the agent. Furthermore, the agent attempts to assign credit or blame for the change upon the objects and situations it recognizes in its environment by recording a preference vector on that object. As in humans and animals, we decide what objects to credit or blame in an emotion-specific fashion (Frijda 1987).
- **Responding to Associations:** When an agent identifies an object or situation it has seen before, it also retrieves the emotional vectors associated with that object or situation. In turn, the agent's situation evaluation system is modified to incorporate the past emotional vector into the current state. When an agent recognizes an object or situation, it can use the preferences and outcomes associated with the object and situation to modify its concern values. This is done in a concern-specific fashion; a safety concern may place different weights on preferences and outcomes than a socialization concern.

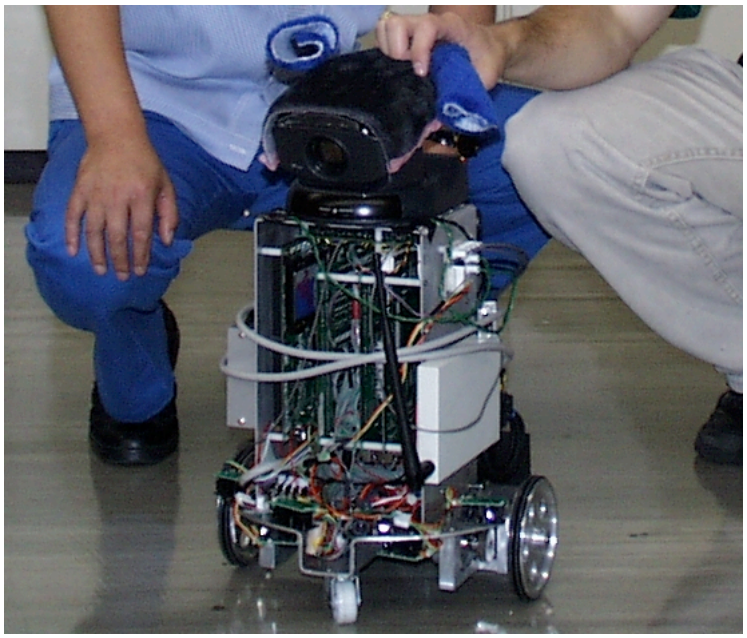

Figure 2: The Yamaha robot prototype

ELTM thus combines ideas like Frijda's (1987) — that emotions represent an agent's concerns about the world — with ideas of case based reasoning (Kolodner 1993) — that learning takes place by remembering the lessons taught by experiences. Emotional adaptation consists of learning from the emotional content of specific past experiences and, over time, from the emotional labeling of objects that have appeared again and again. We represented these emotional experiences as a vector based on a deliberate parallel with Simon's (1983) idea of emotion as the evaluation function of a multidimensional subjective expected utility (SEU) problem.

In this view, emotions break the very difficult problem of deciding between otherwise incommensurable choices into separate problems with different reward functions, each of which can be optimized separately when the emotional system puts it into the forefront. Our goal in doing so was to provide a basis for deploying other learning systems consistent with the SEU framework. For example, interpreting the emotion vector as a multidimensional reward signal was inspired by the LARC model (Santamaria 1997), which used a single reward signal for a case-based reinforcement learning algorithm.

Evaluation of the Approach

We developed two implementations of the PEPE architecture on top of the TeamBots platform (Balch 2000). The first, developed at Georgia Tech for a tracked robot testbed, was used for testing reactive control and facial recognition (Stoytchev & Tanawongsuwan 1998). The second implementation, developed by a joint Georgia Tech-Yamaha team for a wheeled robot prototype (Figure 2), was focused on testing the emotional long term memory system. This prototype had a targetable camera head and two touch sensors, one on the head and one on the "rear" of the robot. At the time of our tests, the face recognition system was not complete, so we gave our volunteers colored t-shirts to enable the robot to recognize them.

To test the PEPE architecture and our ELTM model, we implemented a simple library of behaviors, such as wandering, approach and avoidance, which could in turn be composed into higher level behaviors such as "playing" (alternately wandering and approaching an object) and "fleeing" (backing up, executing a fast 180, and running away). The emotion model extended this with a simple set of concerns, including avoiding pain, which we derived from "kicks" to the rear sensor, and socialization, which we derived from a combination of proximity to people objects and "petting" the head sensor. The robot had several emotional states, including a neutral state, a "happy" state associated with socialization, and a "fearful" state associated with pain. The robot's planner attempted to find plans which matched the current emotional state and execute them. The robot's typical behavior was to wander looking for someone to play with it, and then to attempt to stay close to individuals who "petted" it and to flee from individuals who "kicked" it.

Prior to the addition of the ELTM, the robot had considerable internal flexibility, but externally appeared no more sophisticated than a system with two buttons that switched it between behavior modes for "play" and "run away". The emotional long term memory extended this behavior. Its situation recognizer identified moving patches as distinct objects by color. The outcome associator detected either strong changes to single concerns (e.g., a nearby user deciding to "pet" the robot, strongly increasing the level of the socialization concern) or shifts in the emotional stance (e.g., a user deciding to "kick" the robot, increasing the pain concern and causing the robots stance to switch from "happy" to "fearful"); this shift

triggered updates to the emotional vectors associated with all detected objects, along with the storage of a case representing the robot's current emotion, stance, plan, action, overall environment, and present objects. The situation recognition system continually tried to find relevant cases and to determine whether detected objects had been previously seen; when previously seen objects or situations were detected, these were blended in to the robot's current concern state. We tested this extensively in simulation, and when the robot prototype was next available for testing, in a series of live trials lasting for several days.

The results of adding the memory model were dramatic. Without authoring any additional plans, behaviors or emotional states, the robot's behavior nonetheless changed. Now, its initial wandering behavior was augmented by self-initiated rather than reactive approach-avoidance behaviors: rather than wait for a user to pet it or kick it, it would actively approach users that had petted it in the past or avoid (and sometimes abruptly flee) from users who had kicked it.

After the conclusion of the joint Georgia Tech-Yamaha tests on the Yamaha prototype, the next step was to port the emotion model back to the Georgia Tech testbed, which had a much wider array of sensors and effectors and a much more advanced vision system. However, the project was canceled before this work could be ported back to the testbed, so we could not run more extensive tests on either robot.

Therefore, one of the major goals of the project failed: we cannot report results on how well this model performed on longer human interactions, and the results we can report were conducted in a very small set of trials. While we were satisfied with what we achieved — the emotional long term memory at least *appeared* to make the robot's behavior far more rich based on the same set of base behaviors — by itself this was still impressive primarily in a "look ma no hands" way (McCarthy 1974).

However, in a similar but independent project also sponsored by Yamaha, Velasquez et al (1998) implemented a model of emotional memory in a similar robot called Yuppy. There were differences between these systems – for example, the emotion releasers in the Yuppy model had short term memory, enabling it to "habituate" to constantly present stimuli, and the case library in the PEPE model stored episodes, enabling it to associate emotions with situations — but both incorporated an emotion model, learned emotional responses, and the ability to change behaviors based on the current emotional state.

In a series of trials similar to what we conducted on PEPE, Velasquez demonstrated Yuppy could learn to prefer or fear users based on how the users treated it. This obviously does *not* strengthen the evidence for the specific PEPE model; however, based on the similarities between the PEPE and Yuppy models, we believe the similarity of the results *does* suggest that the general emotional long term memory approach is an effective technique for making agents adapt to users.

Case Study: Emotion-Triggered Behavior Modification for Stable Personalities on the ABL Platform

Embodied agents that interact with humans over longer timeframes should not just adapt to create interest, they should maintain consistent personalities to retain believability. Ideally, we want a self-adapting behavior set for characters, allowing characters to autonomously exhibit their author-specified personalities in new and unforeseen circumstances, and relieving authors of the burden of writing behaviors for every possible situation. In the field of embodied agents, there has been little work on agents that are introspectively aware of their internal state, let alone agents that can rewrite themselves based on deliberating over their internal state. However, this is precisely what psychological research indicates that humans do when stress or disruption makes them aware that their behavior is not living up to their standards.

We propose a model in which emotion provides the agent with the knowledge that its current behavior library is creating inappropriate behavior and should be revised accordingly. For example, a robot pet playing tag with a user would normally succeed using its default chasing behavior, but this might fail if the user is standing on a table, either because the agent has been told not to climb on it as above, or simply because its default chasing behavior does not include jumping. Although the pet can see the user, he cannot reach her, causing the agent's behavior to persistently fail. Our emotion modeling transforms this persistent failure at a given goal into a raised stress level of the agent, which can trigger a behavior modification routine that revises the behavior, for example by adding a jumping behavior to its 'playing tag' repertoire.

A key element to making this behavioral revision work is the use of transformational planning (TP), which does not reason about the domain to generate a plan but instead reasons about a failing plan and transforms it so as to fix the failure without breaking the rest. This insight is key, but we could not directly apply it because TP is generally applied to plans built from STRIPS-like operators, not rich reactive planning languages like ABL. Therefore, we developed novel behavior transformations and techniques for blame assignment that enabled us to apply TP to our behavior modification problem.

Our second case study was implemented on a game scenario which consists of two embodied characters named Jack and Jill. They are involved in a game of Tag where the character who is "It" chases the other around the game area. Each character's behavior library reflects their personality and consists of about 50 behaviors coded in ABL, a language designed for believable characters. Our system (see Figure 2) is composed of a reactive layer which handles the real-time interactions, and a reasoning layer responsible for monitoring the character's state and making repairs as needed.

Reactive Control Using the ABL Programming Language

Our game environment presents a certain set of challenges for the reactive layer. First, a real-time game domain requires the reactive layer to have a fast runtime processing component with a short sense-decide-act loop. Second, the game world's interactive nature entails that the reactive layer handles conditional execution appropriately and provides the ability to support varying behaviors under different situations at runtime. Finally, for game worlds containing embodied, believable

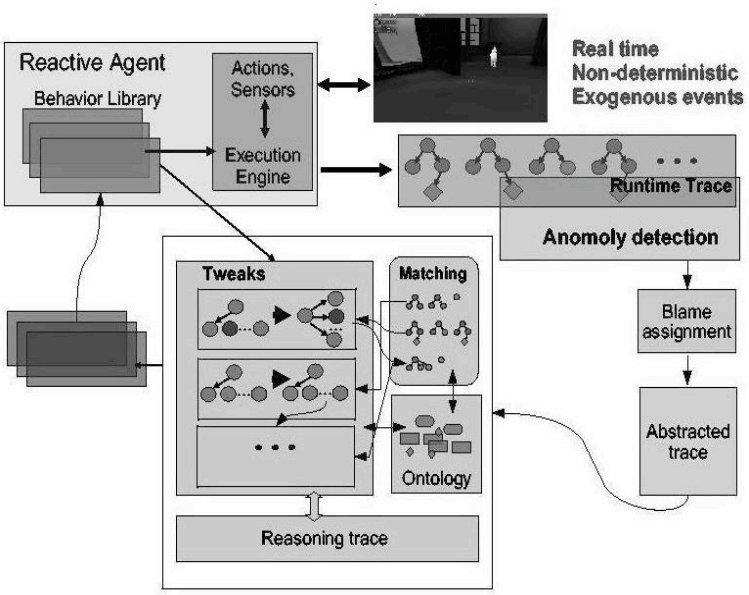

Figure 3: Architecture of the behavior transformation system.

characters, the reactive layer must provide support for the execution of multiple, simultaneous behaviors, allowing characters to gaze, speak, walk around, gesture with their hands and convey facial expressions, all at the same time.

To meet these requirements we use A Behavior Language (ABL) for the reactive layer. ABL is explicitly designed to support programming idioms for the creation of reactive, believable agents (Mateas and Stern, 2004). Its fast runtime execution module makes it suitable for real-time scenarios. ABL is a proven language for believable characters, having been successfully used to author the central characters Trip and Grace for the interactive drama Facade (Mateas and Stern, 2003).

A character authored in ABL is composed of a library of behaviors, capturing the various activities the character can perform in the world. Behaviors are dynamically selected to accomplish goals - different behaviors are appropriate for accomplishing the same goal in different contexts. For example, the goal of expressing anger can be accomplished through either a behavior that screams or a behavior that punches a hole in the wall. Behaviors consist of sequential or parallel steps; steps can be subgoals, mental updates, or game actions.

Currently active goals and behaviors are captured in the active behavior tree. During execution, steps may fail (e.g., no behavior can be found to accomplish a subgoal, or an action fails in the game world), potentially causing the enclosing behavior to fail. Step and behavior annotations can modify the cascading effects of success and failure. Behavior preconditions are used to find appropriate behaviors for accomplishing a goal in the current context. Conditions test against working memory, which encodes both currently sensed information and agent-specific internal state (e.g., emotional state).

ABL's runtime execution module acts as the front-end for communication with the game environment. It constantly senses the world, keeps track of the current game state, updates the active behavior tree and initiates and monitors primitive actions in the game world. Continuously monitored conditions, such as context conditions and success tests, provide immediate, reactive response. Furthermore, the runtime system provides support for meta-behaviors that can monitor (and potentially change) the active behavior tree.

Implementing Emotion-Driven Behavior Modification

To support emotion-driven behavior modification, we implemented a reasoning layer that supports anomaly detection, blame assignment, and behavioral modification. Anomaly detection tracks long-term patterns in the character's behavior execution and detects violations of the author-specified behavior contract. When a contract violation is detected, blame assignment uses the execution trace to identify one or more behaviors that should be changed. The behavioral modification component repairs offending behaviors identified during blame assignment and reloads them into the agent.

- **Anomaly Detection:** Authors need a way to specify contracts about long-term character behavior; when the contract is violated, the reasoning layer should modify the behavior library. Our emotion model is an OCC model based on Em (Reilly 1996). Emotion values serve as compact representations of long-term behavior: a character's emotional state is modified when behaviors succeed or fail in a way defined by the author as part of specifying the character personality. The author specifies personality-specific constraints on behavior by specifying bounds for emotion values. The reasoning layer interprets an emotion exceeding its bounds to mean that the current behavior library is creating inappropriate long-term behavior and that it should seek to assign blame and change the behavior.

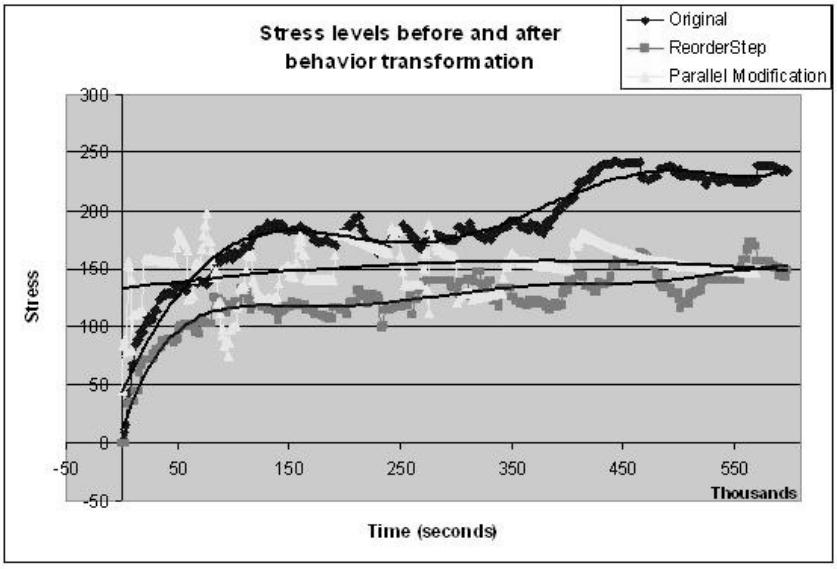

Figure 4: Average stress level from the evaluation experiment

- **Blame Assignment:** The behaviors that should be revised in response to a violation of the personality contract are determined using the meta-reasoning capability of ABL to trace agent execution. Blame assignment analyzes the past execution trace and identifies the behavior with the maximal contribution to the out-of-bound emotion value, amortized over time, as the responsible behavior.
- **Behavior Modification:** Offending behaviors are modified using a set of modification operators. Applicability of an operator depends on the role the behavior plays in the execution trace — that is, on the explanation of how the behavior contributed to a contract violation. Modification operators are categorized according to failure patterns, which provide an abstraction mechanism over the execution trace to detect the type of failure that is taking place. Failure patterns are encoded loosely as finite state machines that look for patterns in the execution trace.

At runtime, the system detects when the author-provided behavior contract has been violated. Once blame assignment has determined the offending behavior, the system uses the failure patterns to explain the behavior's role in the contract violation. The set of matching failure patterns provide an associated set of applicable behavior modification operators to try on the offending behavior, which are tried one at a time until one succeeds. We then modified ABL's runtime system and compiler so that modified behaviors can be compiled and reloaded into the agent, allowing the game to continue uninterrupted.

Evaluation of the Behavior Transformation Architecture

We evaluated our behavior adaptation system on Jack and Jill, two hand-authored embodied characters designed to play a game of Tag. Each character has its own personality that affects the way they approach play: Jack likes to daydream and is not particularly interested in the game, whereas Jill is bored if she is not being chased or chasing someone. Jack and Jill were initially authored by people on a different research project, providing a great opportunity for us to evaluate our system. Their behavior set made fixed assumptions about world dynamics which will be ineffective at maintaining personality invariants in the face of change. If our system can help maintain those invariants then it is an effective means of behavior adaptation.

Specifically, we provided emotion annotations by associating a stress emotion with being chased and placing nominal bounds on stress, specifying a contract on Jack's intended personality. We then tested whether our system is able to successfully modify the behavior library to changing environments. In our experiment, we simulated a changing world by moving the tag agent whose behaviors had been built for a specific map into a larger and sparser version.

Our experimental procedure involves first running the game scenario without the adaptation mechanisms and continuously observing Jack's stress level. We then run Jack with the adaptation mechanisms. Figure 4 shows Jack's stress levels averaged over five 10-minute games before adaptation, and with two different behavior libraries modified by our system. Blame assignment found that the behavior *Run_Away_1* is responsible for stress exceeding bounds. In the ideal case, Jack would run away for a while, until he was able to escape out of sight, at which point, he would head for a hiding place. Trace analysis however shows that Jack turning around to ensure he is not being followed always fails. Jack is never able to run away and escape out of sight long enough to risk going to a hiding place. This situation tends to occur on our test maps because they are sparse; with fewer obstacles it is more difficult for Jack to ever escape out of sight. As a result, Jack is continuously under immediate pursuit and his stress level quickly exceeds bounds.

In our runs, the behavior adaptation system found two different modifications that brought stress back in bounds. In the

```
sequential behavior AvoidItPerson() {
precondition {(ItWME itPlayerName :: itAgent)
        !(AgentPositionWME objectID == itAgent)}
    with(post) subgoal Hide();
    with(post) subgoal TurnAroundEnsureEscape();}
```

Figure 5: The modified behavior

first case, the system changed the *AvoidItPerson_3* behavior (see Figure 5) from a sequential behavior to a parallel behavior. Originally we expected Jack to ensure no one is following before hiding, but the system's change is actually quite reasonable. When pressed, it makes sense to keep running while turning around. If it turns out someone is following you, you can always change course and not go to the secret hiding place. Visually, this change was quite appealing. Jack, when running away, would start strafing towards his hiding place, allowing him to move towards his destination while keeping a look out.

Unfortunately, this change was unstable. Due to how Jack navigates, if he cannot see his next navigation point, he will stall (a defect in his navigation behaviors). Surprisingly, even with this defect, Jack with this change is able to stay within his normal stress bounds. We initially assumed this was because the defect happened rarely, but in fact it was the opposite. While running away, Jack was always getting stuck, allowing Jill to tag him. This decreases stress because Jack is not as stressed when he is the pursuer; he can take his time and is not pressed.

At one level this result is surprising and wonderful. Jack, the daydreamer, successfully "gamed" the system: as "It" he does not have to work so hard to play. But for the purpose of our evaluation, this change was nevertheless undesirable. Jack is violating an implicit behavior contract that Jack should not allow himself to be tagged. The adaptation system essentially found a clever way to take advantage of the under specification of the author's intent. After amending the specifications, our behavior adaptation system found an alternate change: to reorder the steps inside *AvoidItPerson_3*. In the new behavior set, *AvoidItPerson_3* first hides and then turns around to ensure no one is following instead of the other way around. This results in behavior as good if not better than the parallel version.

Future Trends

Our case studies were implemented independently on two different systems, but the underlying reactive control systems and the role of the emotion models used were very similar. Therefore, the natural next step would be to attempt to incorporate both models into a single system with both emotional adaptation and personality updates.

In the PEPE system, the emotion vector was relatively simple, as were the learning algorithms. Another natural next step would be to use a richer emotion model, such as the Em model used in our ABL work, or to use more sophisticated learning algorithms, such as a full version of the Santamaria LARC model that PEPE's learning model emulated.

In our ABL work, to increase the transformational power of our system we are adding more behavior modification operators, which has several effects. First, as the number of operators increases, the time required to reason about them and find the applicable set increases. Second, operators for more complex scenarios may have a lower success rate, requiring us to focus the search through behavior transformation space. It will become necessary for the reasoning layer to learn which operators are best applicable in which situations, such that fewer operators have to be tried. These characteristics of the problem make a case-based approach, as a form of speedup learning, very attractive.

Conclusion

Unlike a psychological model of emotion, which can be tested against the behavior of humans and animals, evaluating the performance of an artificial intelligence system that displays or uses emotion is difficult. Our results with PEPE and ABL are suggestive but it is difficult to prove that they are "better" than a hand-authored system. However, our experiences developing PEPE nonetheless did teach a few important lessons:

- **Layered Architectures Aid Behavioral Authoring:** For a variety of reasons we were only able to use part of the Georgia Tech PEPE code to develop the joint prototype; nonetheless, by layering the system we were able to achieve a large amount of work in a short period of time. The TeamBots system provided a set of reactive behaviors, upon which we layered the planner, the emotion system, and the memory system; furthermore it insulated the higher levels of the system from the robot implementation, enabling us to test the behaviors in simulation while physical issues were worked out on the robot testbed. The PEPE portion of the joint testbed took approximately two man-months of programmer effort, and the complete software developed for these tests, including low-level control and the visual processing system, took approximately six man-months.
- **Emotional Adaptation Increases Behavioral Flexibility:** The emotional memory system dramatically changed

the external behavior with no significant changes to the existing plans, requiring only the incorporation of remembered emotion values into the current emotional state. This simple shift changed the appearance of the robot's behavior from a creature that reacted in a fixed way to external contact from an end user to a creature that had internally generated behaviors that could be affected by the sight of a person from a distance.

- **Active Reminding Is Required for Emotional Adaptation:** The emotion model changed the agent's personality, but its behavior appeared fixed, similarly, just storing cases in the robot's case library did not change its behavior. The critical enabler for emotional memory was the situation recognizer: to learn, the robot required an active reminding process constantly trying to identify objects and situations in terms of its experience. The emotion system only had the opportunity to adapt when a case was retrieved or an object identified.

Similarly, developing the behavior transformation system for ABL also taught a few important lessons:

- **Transformational Planning Aids Behavior Transformation:** In an interactive, real-time domain, characters are constantly interacting with the user. Furthermore, their actions are non-deterministic, need to be executed in parallel, and have effects that are difficult to quantify. Transformational planning made it possible for our characters to modify their behaviors, but we had to develop novel behavior transformations and techniques for blame assignment to enable TP to deal with this complexity and non-determinism.
- **Language Support for Behavior Transformation Makes Modifying Behavior Easier**: Behaviors written for a believable character are code written in a programming language, so modifying behaviors at run time involves rewriting the code for the agent. We spent a lot of time implementing behavior modification operators to accomplish this. Authoring behaviors in a programming language that provided native support for changing its functional constructs (e.g., treating functions as first order entities as in LISP) would have made modifying the behaviors at run-time a much easier process.

Our work used emotion as a trigger for learning about the environment and agents within it, and as a trigger for behavioral change. This model made it possible for us to develop sophisticated and sometimes surprising agent behaviors in less time and with less effort than we could have done otherwise. Therefore, we conclude that emotion-driven learning and emotion-driven behavioral updates are useful methods for developing believable agents that adapt to their environments and users in a way which appears emotionally plausible while maintaining a consistent personality.

References

Alpaydin, E. (2004). *Introduction to machine learning*. Cambridge, MA: The MIT Press.

Anderson, J.R. (2000). *Learning and memory: an integrated approach.* New York, NY. John Wiley & Sons.

Arkin, R.C. (1989). Motor schema-based mobile robot navigation. *The International Journal of Robotics Research*, Vol. 8, No. 4, pp. 92-112.

Balch, T. (2000). TeamBots software and documentation. Retrieved May 18, 2008 from http://www.cs.cmu.edu/~trb/TeamBots/.

Bartlett, F.C. (1932). Remembering. Cambridge: Cambridge University Press.

Bartneck, C. (2002). Integrating the OCC Model of Emotions in Embodied Characters. *Proceedings of the Workshop on Virtual Conversational Characters: Applications, Methods, and Research Challenges*, Melbourne.

Benjamin, M., McKeachie, W., Lin, Y.-G., & Holinger, D. (1981). Test anxiety: deficits in information processing. *Journal of Educational Psychology*, 73, pp. 816-824.

Brooks, R. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation* 2 (1): pp. 14-23.

Caprara, G.V. & Cervone, D. (2000). *Personality: determinants, dynamics and potentials.* Cambridge University Press.

Champanard, A.J. (2007). Evolving with Creatures' AI: 15 Tricks to Mutate into Your Own Game. AIGameDev.com Reviews, October 21, 2007. Retrieved May 17, 2008 from http://aigamedev.com/reviews/creatures-ai.

Cyberlife Technology Ltd. (1997). *Creatures.* Computer game, multiple platforms, published by Virgin Interactive Entertainment.

Damasio, A. (2000). A second chance for emotion. In R. D. Lane & L. Nadel (Eds.), *Cognitive neuroscience of emotion.* pp. 12-23. New York: Oxford University Press.

Elliott, C. D. (1992). *The affective reasoner: a process model of emotions in a multi-agent system.* Unpublished Ph.D. thesis. Evanston, IL: The Institute for the Learning Sciences, Northwestern University.

Frijda, N. (1986). *The Emotions.* New York, NY: Cambridge University Press.

Gluck, M., Mercado, E., & Myers, C. (2008). *Learning and Memory: From Brain to Behavior.* New York, NY: Worth Publishers.

Haist, F., Gore, J.B. & Mao, H. (2001). Consolidation of memory over decades revealed by functional magnetic resonance imaging. *Nature Neuroscience,* 4, pp.1139-1145.

Huebner, R. (2000). Postmortem: Nihilistic Software's Vampire: The Masquerade — Redemption. *Game Developer Magazine*, July 2000, pp. 44-51.

Johnston, O & Thomas, F. (1995). *Disney Animation: The Illusion of Life.* New York, NY: Hyperion.

Karunaratne, S. & Yan, H. (2001, May). Interactive emotional response computation for scriptable multimedia actors. In: *Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing*, Hong Kong, China.

Koda, T. (1996). *Agents with faces: a study on the effect of personification of software agents.* Unpublished Masters Thesis. Cambridge, MA: MIT Media Lab, Massachusetts Institute of Technology.

Kolodner, J.L. (1984). *Retrieval and organization strategies in conceptual memory: a computer model.* Northvale, NJ: Lawrence Erlbaum Associates.

Kolodner, J.L. (1993). *Case-Based Reasoning.* San Francisco, CA: Morgan Kaufmann.

LeDoux, J. (1996). *The emotional brain: The mysterious underpinnings of emotional life*. New York, NY: Simon & Schuster.

Li, T., Ma, Y., Qiu, Y., & Yue, P. (2007, February). Modeling Personality, Emotion and Mood for a Pedagogical Agent. In *Proceedings of Artificial Intelligence and Applications 2007*, Innsbruck, Austria. Calgary AB, Canada: Acta Press.

Loyall, A. B. (1997). *Believable Agents: Building Interactive Personalities.* Ph.D. Thesis. Technical Report CMU-CS-97-123, Pittsburgh, PA: School of Computer Science, Carnegie Mellon University.

Mateas, M. & Stern, A. (2003, March). Facade: An Experiment in Building a Fully-Realized Interactive Drama. Paper presented at the Game Design Track of the Game Developer's Conference, San Jose, CA.
Mateas, M. & Stern, A. (2004). A Behavior Language: Joint action and Behavioral Idioms. In Prendinger, H. and Ishizuka, M., (Eds.), *Life-Like Characters. Tools, Affective Functions, and Applications. Cognitive Technologies*. Berlin Heidelberg: Springer Verlag.
Maxis Software, Inc. (2000). *The Sims*. Computer game, PC, published by Electronic Arts, Inc.
McCarthy, J. (1974). Review of "Artificial Intelligence: A General Survey". Artificial Intelligence, Vol. 5, No. 3 (1974). Retrieved May 18, 2008 from http://www-formal.stanford.edu/jmc/reviews/lighthill/lighthill.html.
McGaugh, J. (2003). *Memory and emotion: the making of lasting memories*. New York, NY: Columbia University Press.
Millington, I. (2006). *Artificial Intelligence for Games*. San Francisco, CA: Morgan Kaufmann.
Minsky, M. (2007). *The Emotion Machine: Commonsense Thinking, Artificial Intelligence, and the Future of the Human Mind*. New York, NY: Simon & Schuster.
Mitchell, T. (1997). *Machine learning*. McGraw Hill.
Reilly, W. S. N. (1996). *Believable Social and Emotional Agents*. Unpublished Ph.D. Thesis. Pittsburgh, PA: Carnegie Mellon University.
Ohlen, J., Zeschuk, G & Muzyka, R. (2001). Postmortem: BioWare's Baldur's Gate II. *Game Developer Magazine*, March 2001, pp. 54-66.
Ohman, A., Flykkt, A. & Lundqvist, D. (2000). Unconscious emotion: Evolutionary perspectives, psychophysiological data and neuropsychological mechanisms. In R. D. Lane & L. Nadel (Eds.), *Cognitive neuroscience of emotion*. pp. 296-327. New York: Oxford University Press.
Ortony, A., Clore, G.L. & Collins, A. (1988). *The Cognitive Structure of Emotions*. Cambridge University Press.
Paiva A., Dias, J., Sobral, D., Aylett, R., Woods, S., Hall, L. & Zoll, C. (2005). Learning by feeling: evoking empathy with synthetic characters. *Applied Artificial Intelligence*, Volume 19, Numbers 3-4, March-April 2005 , pp. 235-266(32). Taylor and Francis Ltd.
Peot , M. & Smith, D. (1992). Conditional Nonlinear Planning. In *Artificial Intelligence Planning Systems: Proceedings of the First International Conference*, June 15-17, 1992, College Park, Maryland. San Francisco, CA: Morgan Kaufmann.
Purdy, J.E., Markham, M.R., Schwartz, B.L., & Gordon, W. (2001). *Learning and Memory*. Belmont, CA: Wadsworth.
Ruys, K.I. & Stapel, D.A. (2008). The Secret Life of Emotions. *Psychological Science*, Volume 19, Number 4, April 2008 , pp. 385-391. Blackwell Publishing.
Saltzman, M. (Ed.) (1999). *Game Design: Secrets of the Sages*. Indianapolis, IN: Brady Publishing.
Santamaria, J.C. (1997). Learning adaptive reactive agents. Unpublished doctoral dissertation. Atlanta, GA: Georgia Institute of Technology.
Schwab, B. (2004). *AI Game Engine Programming*. Hingham, Massachusetts: Charles River Media.
Simon, H.A. (1983). *Reason in Human Affairs*. Stanford, CA: Stanford University Press.
Spector, W. (2000). Postmortem: Ion Storm's Deus Ex. *Game Developer Magazine*, November 2000, pp. 50-58.
Standifer, C. (1995). Personal communication.
Stoytchev, A. & Tanawongsuwan, R. (1998, July). Pepe: PErsonal PEt. In *Video Proceedings of the AAAI-98 Mobile Robot Exhibition*, Madison, WI.
Studdard, P. (1995). *Representing human emotions in intelligent agents*. Unpublished Masters Thesis. Washington, DC: The American University.
Sutton, R.S. & Barto, A.G. (1998). *Reinforcement learning: an introduction*. Cambridge, MA: MIT Press.
Tozour, P. (2002). The evolution of game AI. In S. Rabin (Ed.), *AI Game Programming Wisdom*. Hingham, MA: Charles River Media.
Tulving, E. & Craik, F.I.M. (2000). *The Oxford Handbook of Memory*. New York, NY: Oxford University Press.
Weld, D. S., Anderson, C.R. & Smith, D.E. (1998). Extending Graphplan to Handle Uncertainty & Sensing Actions. In *Proceedings of AAAI 1998*. AAAI Press.
Winkielman , P. & Berridge, K. (2004). Unconscious Emotion. In *Current Directions in Psychological Science*, Volume 13 #3, pp. 120-123. Blackwell Synergy.
Woodcock, S. (2000a). *AI Roundtable Moderator's Report*. Report on the AI Roundtable of the Game Developer's Conference, San Jose, CA. Retrieved May 17, 2008 from http://www.gameai.com/cgdc00notes.html.
Woodcock, S. (2000b). Game AI: The State of the Industry. *Game Developer's Magazine*, August 2000, pp. 24-32.

Key Terms

The following terms and acronyms were used in this chapter and might warrant definitions in the Handbook:

ABL: A Behavior Language
ABL is a programming language explicitly designed to support programming idioms for the creation of reactive, believable agents (Mateas and Stern, 2004). ABL has been successfully used to author the central characters Trip and Grace for the interactive drama Facade (Mateas and Stern, 2003). The ABL compiler is written in Java and targets Java; the generated Java code is supported by the ABL runtime system.

Appraisal In the OCC (Ortony et al. 1988) and Frijda (1993) models of emotion, appraisal matches the experience of an agent against its goals, standards, preferences and other concerns. The results of this matching give emotional events their positive or negative feeling or weight, called affect, and can also place this affective response in context.

Blame Assignment In learning and adaptation, blame assignment is the process of identifying the causes of a failure of a computational system to deliver the behaviors desired of it.

Case-Based Reasoning Case-based reasoning (Kolodner 1993) is a reasoning architecture that stores experiences with lessons learned as cases in a case library and solves problems by retrieving the case most similar to the current situation, adapting it for reuse, and retaining new solutions once they have been applied. Case-based reasoning is also a pervasive behavior in everyday human problem solving.

Concern In Frijda's (1986) model of emotion, concerns correspond to the needs, preferences and drives of an agent – things that "matter" and can trigger changes to the emotional state of the agent.

OCC Model of Emotion Ortony, Clore and Collins's (Ortony et al. 1988) model of emotion is a widely used model of emotion that states that the strength of a given emotion primarily depends on the events, agents, or objects in the environment of the agent exhibiting the emotion. A large number of researchers have employed the OCC model to

generate emotions for their embodied characters. The model specifies about 22 emotion categories and consists of five processes that define the complete system that characters follow from the initial categorization of an event to the resulting behavior of the character. These processes are namely a) classifying the event, action or object encountered, b) quantifying the intensity of affected emotions, c) interaction of the newly generated emotion with existing emotions, d) mapping the emotional state to an emotional expression and e) expressing the emotional state.

MDP: Markov Decision Processes Markov decision processes provide a mathematical framework for modeling decision-making characterized by a set of states where in each state there are several actions from which the decision maker must choose and transitions to a new state at time $t + 1$ from time t are only dependent on the current state and independent of all previous states. MDPs are useful for studying a wide range of optimization problems solved via dynamic programming and reinforcement learning.

SEU: Subjective Expected Utility Theory Subjective expected utility theory (Simon 1983) holds that a rational agent should attempt to maximize its reward by choosing the action with the highest *expected utility* — effectively, the sum of the rewards of the outcomes discounted by the probabilities of their occurrence.